WEBMESTRE : CONCEPTION DE SITES ET ADMINISTRATION DE SERVEURS WEB

Installation et administration d'un serveur web

Module 25793 TP A5 (1/2 valeur)

CONSERVATOIRE NATIONAL DES ARTS ET METIERS

Chapitre 13

Analyse des fichiers journaux
Evaluation des accès au serveur web

Le plus grand soin a été apporté à la réalisation de ce support pédagogique afin de vous fournir une information complète et fiable. Cependant, le Cnam Champagne-Ardenne n'assume de responsabilités, ni pour son utilisation, ni pour les contrefaçons de brevets ou atteintes aux droits de tierces personnes qui pourraient résulter de cette utilisation.

Les exemples ou programmes présents dans cet ouvrage sont fournis pour illustrer les descriptions théoriques. Ils ne sont en aucun cas destinés à une utilisation commerciale ou professionnelle.

Le Cnam ne pourra en aucun cas être tenu pour responsable des préjudices ou dommages de quelque nature que ce soit pouvant résulter de l'utilisation de ces exemples ou programmes.

Tous les noms de produits ou autres marques cités dans ce support sont des marques déposées par leurs propriétaires respectifs.

Ce support pédagogique a été rédigé par Michel Melcior, enseignant vacataire au Cnam Champagne-Ardenne.

Copyright © 2001-2003 Centre d'Enseignement A Distance du Cnam Champagne-Ardenne.

Tous droits réservés.

Toute reproduction, même partielle, par quelque procédé que ce soit, est interdite sans autorisation préalable du Cnam Champagne-Ardenne. Une copie par xérographie, photographie, film, support magnétique ou autre, constitue une contrefaçon passible des peines prévues par la loi, du 11 mars 1957 et du 3 juillet 1995, sur la protection des droits d'auteur.

ANALYSE DES FICHIERS JOURNAUX, EVALUATION DES ACCES AU SERVEUR WEB.

1. OBJECTIFS

En fin de séance, vous devriez être capable de :

- Gérer les fichiers journaux du système et d'analyser les accès au serveur web avec l'outil Webalizer.

2. INTRODUCTION

L'analyse régulière et l'archivage des divers fichiers journaux du système est une tâche courante et essentielle de l'administrateur système.

Les fichiers journaux sont situés dans **`/var/log`**.

On peut citer le fichier messages qui contient les événements enregistrés par le système. En voici un extrait à titre d'exemple sur un serveur appelé "tango":

```
Feb  13  17:25:42  tango  mountd[524]:  authenticated  mount  request  from
mairie.iutgtr-chalons.com.fr:613 for /u (/u)

Feb  13  17:28:04  tango  mountd[524]:  authenticated  unmount  request  from
fagnieres.iutgtr-chalons.com.fr:625 for /u (/u)

Feb  13  19:34:02  tango  mountd[524]:  authenticated  mount  request  from
stgibrien.iutgtr-chalons.com.fr:604 for /u (/u)

Feb  13  19:54:57  tango  mountd[524]:  authenticated  unmount  request  from
omey.iutgtr-chalons.com.fr:719 for /u (/u)
```

On y trouve consigné: La date, le nom du serveur, le nom et le pid (process identifier) du processus ayant émis un message (ici mountd, 524) et le message émis par le processus.

Vous vous rendrez vite compte que la taille de ce fichier prend rapidement de l'ampleur.

Comment rechercher et pister un événement dans ce fichier ?

La commande **`cat`** permet de visualiser à l'écran le contenu d'un fichier. Vous pouvez vous en servir pour prendre connaissance du fichier **`/var/log/messages`** .

Si vous ne souhaitez voir que certains événements, il faudra filtrer la sortie de cat avec une commande appelée **`grep`** .

`grep` filtre le flux de lignes de texte qui lui est envoyé en n'affichant les lignes qui contiennent des éléments contenus dans une chaîne de caractères de filtrage appelée aussi "**expression régulière**[1]".

[1] **Voir annexe : expressions régulières.**

Pour diriger la sortie de **`cat`** vers l'entrée de **`grep`** il faut créer un tube (pipe) entre les deux processus. Ce tube s'obtient avec le caractère **|** (**Alt Gr** + **touche 6** au dessus de T et Y)

Un exemple :

`cat messages | grep "mountd"` n'affichera que les lignes contenant la chaîne mountd .

L'écriture d'expressions régulières alambiquées est un sport de prédilection des administrateurs UNIX.

Exercice :

- Affichez uniquement les lignes du fichier **/var/log/messages** concernant cette journée

Vous pouvez avoir besoin de stocker le résultat dans un fichier pour une consultation ultérieure. Il suffit de rediriger la sortie de grep vers le fichier avec le caractère **>** .

Exemple : cat messages | grep "mountd" > rapport.log

Pour information, les deux caractères **>>** redirigent en mode ajout (en fin d'un fichier), le caractère **>** seul redirige en mode écrasement (on remplace le contenu d'un fichier).

Exercice :

- Redirigez le résultat précédent vers un fichier **essai.log** dans votre répertoire personnel (astuce, le répertoire personnel se note ~. Par exemple, si vous êtes **root**, **~/logs** est équivalent à **/root/logs**).

Vos tâches d'administrateur consistent à :

Surveiller l'accroissement de ces fichiers et éviter qu'ils ne monopolisent l'espace disque. Eventuellement, vous devez compresser ou détruire certains fichiers journaux.

Analyser régulièrement leur contenu pour surveiller le fonctionnement du système.

Archiver (support externe) le contenu de certains fichiers journaux dont le contenu ne doit pas être perdu.

Un outil est à votre disposition pour vous aider. Il s'agit de l'utilitaire **`logrotate`**[2].

Exercice :

- Parcourez l'annexe sur logrotate.
- Déterminez le nom du répertoire inclus dans le fichier de configuration **/etc/logrotate.conf** ?
- Quel fichier contient les réglages relatifs à Apache ?
- Quel est le nombre de rotations prévu pour les fichiers journaux d'Apache ?

Logrotate utilise le programme de compression **`gzip`** pour compresser les fichiers journaux.

Il faut savoir que **`gzip`** ne permet de compresser qu'un seul fichier à la fois et remplace le fichier non compressé par sa version compressée.

[2] **Voir annexe : logrotate**.

Usage :

gzip monfichier

Va compresser **monfichier** et le remplacer par sa version compressée **monfichier.gz**

gzip -d monfichier.gz

Va décompresser **monfichier.gz** et le remplacer par sa version non compressée.

Le suffixe **.gz** ajouté par **gzip** n'est pas obligatoire.

Comme **gzip** n'est pas capable de compresser plusieurs fichiers à la fois, il est souvent combiné avec la commande **tar** .

tar permet de regrouper plusieurs fichiers en un seul (tarage).

L'usage combiné de **tar** et **gzip** est possible :

tar cvzf mesfichier.tgz fichier1 fichier2 ...

Tare puis compresse **fichier1 fichier2 ...** en un seul fichier: **mesfichiers.tgz**

tar xvzf mesfichiers.tgz

Détare et décompresse.

3. LES FICHIERS JOURNAUX D'APACHE.

L'emplacement des fichiers journaux d'Apache est contrôlé par les directives **ErrorLog** et **CustomLog** du fichier de configuration (**/etc/httpd/conf/httpd.conf**).

3.1. LA DIRECTIVE ERRORLOG

Syntaxe : **ErrorLog nomfichier**
Défaut : **ErrorLog logs/error_log** (Unix)
Défaut : **ErrorLog logs/error.log** (Windows et OS/2)
Contexte : configuration serveur, hôtes virtuels
Statut : noyau

Cette directive définit le nom du fichier dans lequel le serveur marque la trace des erreurs rencontrées. Si le nom de fichier ne commence pas par un slash (/), alors la partie "chemin d'accès" est considérée relativement à **ServerRoot**.

Exemples :

```
ErrorLog logs/error_log

ErrorLog /dev/null
```

Cette dernière expression a pour effet de désactiver la trace d'erreurs.

3.2. LE MODULE MOD_LOG_CONFIG

Le module **mod_log_config** remplace le module **mod_log_common** à partir de la version 1.2. Avant la version 1.2, **mod_log_config** était un module optionnel. Il permet l'enregistrement de la trace des requêtes reçues par le serveur, sous le format Common Log Format, ou un format personnalisé par l'administrateur.

Résumé :

Trois directives sont définies par ce module : **TransferLog** pour créer un fichier de trace, **LogFormat** pour définir un format personnalisé, et **CustomLog** pour définir à la fois un fichier de trace et un format personnalisé en une seule opération. Les directives **TransferLog** et **CustomLog** peuvent être utilisées plusieurs fois dans chaque définition de serveur pour obliger la requête a être tracée dans plusieurs fichiers différents (éventuellement sous plusieurs formats différents).

Formats de fichiers de trace :

Sauf mention contraire exprimée par une directive **LogFormat**, les fichiers de trace crées par **TransferLog** le seront conformément au standard "**Common Log Format**" (**CLF**). Le contenu de chacune des lignes d'un fichier CLF est explicité ci-après. Alternativement, le format du fichier de trace peut être personnalisé (et si plusieurs fichiers de trace sont utilisés, chacun peut être défini dans un format différent). Les formats personnalisés sont créés par les directives **LogFormat** et **CustomLog**.

3.2.1. Common Log Format :

Les fichiers de format "**Common Log Format**" (**CLF**) contiennent des lignes d'enregistrement, à raison d'une ligne par requête. Une ligne est composée de plusieurs champs séparés par des espaces :

```
hôte identité utilisateur_authentifié date requête état volume
```

Lorsqu'un champ n'a pas de valeur particulière, alors il est réduit à un tiret (-). La signification et les valeurs typiques de ces champs sont les suivantes :

hôte

Le nom de domaine entièrement qualifié du client, ou son adresse IP si le nom de domaine n'est pas disponible.

identité

Si l'option IdentityCheck est activée et la machine cliente exécute identd, alors ce champ donne l'information d'identification donnée par le client.

utilisateur_authentifié

Si la requête visait un document protégé par mot de passe, alors ce champ contient le nom d'utilisateur utilisé lors de l'authentification.

date

La date et l'heure de la requête, dans le format suivant :
date = [jour/mois/an:heure:minute:seconde zone]
jour = 2*digit
mois = 3*lettre
an = 4*digit
heure = 2*digit

minute = 2*digit
seconde = 2*digit
zone = (`+' | `-') 4*digit

requête

La chaîne de requête émise par le client, entre guillemets (").

état

Le code de réponse donné par le serveur pour cette requête.

volume

Le nombre d'octets de l'objet renvoyé au client, hormis toute en-tête.

3.2.2. Custom Log Formats **(formats personnalisés) :**

Le descripteur de format donné par des directives **`LogFormat`** et **`CustomLog`** est une chaîne de caractères. Cette chaîne est enregistrée dans le fichier de trace pour chaque requête arrivant au serveur. Elle peut contenir des caractères littéraux copiés dans les fichiers de trace, et des expressions variables (précédées de '%') remplacées au moment de l'écriture de l'enregistrement par les valeurs correspondantes :

`%...b`:	Octets envoyés, non comptées les en-têtes HTTP.
`%...f`:	Nom de fichier
`%...{Foobar}e`:	Le contenu de la variable d'environnement Foobar (Foobar est un exemple).
`%...h`:	L'hôte distant
`%...{Foobar}i`:	Le contenu du champ d'en-tête Foobar: de la requête reçue par le serveur.
`%...l`:	Identité distante (à partir de identd, si fournie)
`%...{Foobar}n`:	Le contenu d'une note "Foobar" d'un autre module.
`%...{Foobar}o`:	Le contenu du champ d'en-tête Foobar: dans la réponse.
`%...p`:	Le port sur lequel la requête a été servie.
`%...P`:	Le PID du processus fils qui a servi la requête.
`%...r`:	La première ligne de la requête
`%...s`:	Code HTTP. Pour les requêtes redirigées en interne, soit le code pour la requête *originale* --- %...>s pour la dernière redirection.
`%...t`:	Temps, au format CLF
`%...{format}t`:	Temps, selon le format spécifié conformément à la syntaxe de la fonction C UNIX strftime(3).
`%...T`:	Le temps mis pour servir la requête, en secondes.
`%...u`:	Utilisateur distant (à partir de données d'authentification. Peut être vide si %s) vaut 401)
`%...U`:	L'URL requise.
`%...v`:	Le nom du serveur (c-à-d. quel hôte virtuel ?)

La partie '...' peut être vide (ex. "%h %u %r %s %b"), ou préciser les conditions pour lesquelles l'information pourra être enregistrée (ce champ sera remplacé par '-' si ces conditions ne sont pas remplies). Notez qu'aucun caractère d'échappement n'est utilisé dans les chaînes %r, %...i et %...o.

Le format dans lequel sont exprimées les conditions est une liste de codes de réponse HTTP, qui peuvent éventuellement être précédés d'un signe '!' de négation.

Deux exemples :

%400,501{User-agent}i provoquera l'enregistrement du champ d'en-tête User-agent: sur erreur de type 400 et 501 (Requête erronée, Non implémenté) uniquement ;

%!200,304,302{Referer}i enregistre le champ Referer: pour toutes les requêtes auxquelles est répondu un code de réponse autre qu'une réponse normale.

Notez que le **Common Log Format** est représenté par la chaîne "**%h %l %u %t \"%r\" %s %b**", laquelle peut servir de base de départ si vous désirez personnaliser votre trace (ex. en ajoutant des champs supplémentaires à la fin).

Le format de trace combiné/étendu de ttpe NCSA serait représenté par la chaîne "**%h %l %u %t \"%r\" %s %b \"%{Referer}i\" \"%{User-agent}i\"**".

Fichiers de trace multiples

Les directives **TransferLog** et **CustomLog** peuvent apparaître plus d'une fois afin de tracer les requêtes dans plusieurs fichiers sous des formats différents. Chaque requête sera enregistrée dans chacun de ces fichiers dansle format associé à chacune des directives correspondantes.

Utilisation avec des hôtes virtuels

Si une section **<VirtualHost>** ne contient aucune directive **TransferLog** ni **CustomLog**, c'est la trace définie pour le serveur principal qui servira pour cet hôte virtuel. Si elles contiennent une ou plusieurs de ces directives, les requêtes servies par cet hôte virtuel ne seront tracées que dans les fichiers déclarés à l'intérieur de cette section, et à aucun moment dans les traces du serveur principal.

3.3. DIRECTIVE : LOGFORMAT

Syntaxe : **LogFormat format [raccourci]**
Défaut : **LogFormat** "**%h %l %u %t \"%r\" %s %b**"
Contexte : configuration serveur, hôtes virtuels
Statut : Base
Compatibilité : **Les raccourcis ne sont disponibles qu'à partir des versions 1.3 d'Apache.**
Module : mod_log_config

Définit le format du fichier de trace par défaut spécifié dans la directive **TransferLog**. Voir la section sur les formats personnalisés pour plus de détails sur les arguments de l'attribut de format.

Si vous précisez un raccourci pour ce format, vous pourrez spécifier ce format par ce raccourci dans d'autres directives **LogFormat** ou **CustomLog** plutôt que réécrire toute la chaîne de description.

Une directive **LogFormat** qui définit un raccourci n'a aucune autre fonction -- c'est-à-dire, elle ne fait que définir le raccourci, sans appliquer le format à quoi que ce soit, ni l'attribuer au défaut.

Voici, par exemple, les formats définis par défaut dans Apache :

```
LogFormat "%h %l %u %t \"%r\" %>s %b \"%{Referer}i\" \"%{User-Agent}i\"" combined
LogFormat "%h %l %u %t \"%r\" %>s %b" common
LogFormat "%{Referer}i -> %U" referer
LogFormat "%{User-agent}i" agent
```

En faisant référence aux raccourcis **combined common referer** ou **agent** dans la directive **Customlog**, vous utiliserez l'un de ces quatre formats.

Vous pouvez, bien sûr, personnaliser ces réglages de base.

3.4. DIRECTIVE CUSTOMLOG

Syntaxe : **`CustomLog fichier format-ou-raccourci`**
Contexte : configuration serveur, hôtes virtuels
Statut : Base
Compatibilité : Le raccourci n'est reconnu qu'à partir de la version 1.3 d'Apache
Module : mod_log_config

Le premier argument est le nom de fichier de trace. Il est défini exactement comme l'argument de la directive **`TransferLog`** ; c'est-à-dire, comme un chemin absolu, ou relatif par rapport à la racine serveur courante.

L'argument format spécifie le format dans lequel les lignes d'enregistrement seront écrites. Les options disponibles pour le format sont exactement les mêmes que pour la directive **`LogFormat`**. Si le format inclue des espaces littéraux, (ce qui sera pratiquement toujours le cas) alors la chaîne entière devra être mise entre apostrophes.

A partir de la version 1.3 d'Apache, vous pouvez remplacer l'expression exhaustive du format par un raccourci, lequel aura été défini par une directive **`LogFormat`**.

Quelques exemples :

```
CustomLog logs/access_log common
CustomLog logs/referer_log referer
CustomLog logs/agent_log agent
CustomLog logs/access_log combined
```

Exercice :
Quel est le format de traces utilisé dans votre configuration Apache par défaut ?

4. L'ANALYSE DES FICHIERS JOURNAUX D'APACHE.

Un outil intéressant pour l'analyse des fichiers journaux d'Apache s'appelle Webalizer.
Vous pouvez télécharger le paquetage de Webalizer à l'adresse : http://www.rpmfind.net/

Cherchez le paquetage webalizer-2.01.10-14mdk.i586.rpm (mandrake cooker).

Installez le paquetage avec :

```
rpm –ivh webalizer-2.01.10-14mdk.i586.rpm
```

Exécutez la commande : webalizer

Par défaut, webalizer est réglé pour analyser /var/log/httpd/access.conf et sortir ses résultats dans /var/www/html/stats. Vous pouvez changer cela avec le fichier /etc/webalizer.conf.

Webalizer va créer toute une série de pages html qui résument les accès au serveur.

Consultez ces pages.

Nous allons à présent utiliser **`logrotate`** sur les fichiers d'Apache.

Dans **`/etc/logrotate.d`** , éditez le fichier **`apache`**. Repérez la partie concernant **`access_log`** et ajoutez la ligne **`rotate 3`** . Elle permettra de garder trois anciennes versions d'**`access_log`** avant suppression.

1. Faites tourner plusieurs fois les fichiers journaux d'Apache avec :

   `logrotate -f /etc/logrotate.d/apache`.

2. Observez le résultat dans le répertoire **`/var/log/httpd`**

5. RESUME.

Le tableau ci-dessous (**à compléter par vos soins**) résume les notions abordées dans cette séance.

Commande	Fonction
gzip	
tar	
logrotate	
errorlog	
customlog	
logformat	
webalizer	

6. ANNEXE

6.1. LOGROTATE

Syntaxe :

```
logrotate [-dv] [-f|--force] [-s|--state file] config_file+
```

Description :

Logrotate est conçu pour faciliter l'administration des nombreux fichiers journaux créés par le système. Il permet d'automatiser les rotations[3] de ces fichiers, leur compression, leur élimination et l'envoi de leur contenu par email.

Chaque fichier journal peut être traité chaque jour, chaque semaine, chaque mois où lorsqu'il devient trop volumineux.

Normalement, logrotate est exécuté comme une tâche journalière gérée par le deamon crond (le planificateur de tâches d'UNIX). Il n'y aura pas plusieurs rotations par jour sauf si le fichier venait à dépasser des critères de taille ou que l'on force manuellement la rotation avec l'option -f.

On peut préciser plusieurs fichiers de configuration dans la ligne de commande. En cas de conflit, ce sont les fichiers de configuration les plus récents qui prévalent. Normalement, il est d'usage de n'utiliser qu'un seul fichier de configuration qui inclura d'autres fichiers.

Options :

-d : on passe en mode debug ce qui implique d'avoir activé l'option -v . En mode debug, les fichiers journaux ne sont pas modifiés.

-f, --force : cette option force une rotation.

-s, --state <statefile> : spécifie à logrotate le fichier à utiliser pour stocker son état. Le fichier par défaut est : /var/lib/logrotate.status.

--usage : affiche un court mode d'emploi.

Le fichier de configuration :

Logrotate est informé de toutes les tâches à réaliser sur les différents fichiers journaux à partir des fichiers de configuration. En voici un exemple :

```
+3i
# sample logrotate configuration file
errors sysadmin@my.org
compress

/var/log/messages {
    rotate 5
    weekly
    postrotate
        /sbin/killall -HUP syslogd
    endscript
}
```

[3] La rotation d'un fichier journal signifie que le fichier courant se voit attribuer un chiffre comme suffixe et qu'un nouveau fichier vide est créé. Par exemple, **journal.log** devient **journal.log.1** et un journal.log vide est créé. Le niveau de rotations peut être supérieur à 1. Si on reprend l'exemple, à la rotation suivante, **journal.log.1** devient **journal.log.2**, journal.log devient journal.log.1 et un nouveau journal.log vide est créé. Il est ainsi possible d'archiver périodiquement des fichiers journaux (logfiles).

```
"/var/log/httpd/access.log" {
    rotate 5
    mail www@my.org
    errors www@my.org
    size=100k
    postrotate
        /sbin/killall -HUP httpd
    endscript
}
```

Les premières lignes sont des options globales. Elles indiquent que :

- Chaque erreur rencontrée au cours des rotations doit être notifiée par email à l'adresse sysadmin@my.org.
- Les fichiers journaux sont compressés après la rotation.

Notez que les commentaires sont précédés en début de ligne par le caractère "# ".

La section suivante du fichier de configuration définit comment traiter le fichier journal `/var/log/messages`. Le fichier journal subira 5 rotations avant d'être effacé. Après chaque rotation (mais avant la compression), la commande `/sbin/killall -HUP syslogd` est exécutée. Ceci a pour effet de relancer le deamon `syslogd` chargé d'enregistrer les messages du système dans `/var/log/messages` et donc de créer un nouveau fichier `/var/log/messages` vide.

La section suivante définit le traitement de `/var/log/httpd/access.log`. La rotation sera effectuée si le fichier dépasse une taille de 100ko et une copie du fichier sera envoyée, non compressée, par email à www@my.org. Après 5 rotations le fichier sera détruit. De plus, chaque erreur rencontrée pendant le processus de rotation sera notifiée par email à l'adresse www@my.org (remarquez que cette directive locale remplace la directive globale). Le double guillemet utilisé en début de section permet de définir un fichier journal comportant des espaces dans son nom. La commande `/sbin/killall -HUP httpd` permet de relancer le deamon httpd et de créer un nouveau fichier `/var/log/httpd/access.log` vide.

Voici des informations complémentaires sur les directives qui peuvent être incluses dans un fichier de configuration de logrotate.

`compress` : après la première rotation, le fichier est compressé avec gzip. Voir aussi nocompress.

`daily` : la rotation s'effectue chaque jour.

`errors` `[addresse email]` : chaque erreur est notifiée à l'adresse précisée.

`ifempty` : la rotation est effectuée même si le fichier journal est vide (c'est l'option par défaut).

`include` `[fichier ou répertoire]` : le fichier spécifié va être inclus dans le fichier de configuration. Cette option permet de découper et d'améliorer la lisibilité de l'ensemble des paramètres. Si un répertoire est donné comme argument, les fichiers contenus dans ce répertoire seront lus et appliqués.

`mail address` : when a log is rotated out-of-existence, it is mailed to address. If no mail should be generated by a particular log, the nomail directive may be used.

`missingok` : si le fichier journal est manquant, logrotate l'ignore et passe à la suite de son travail sans émettre de message d'erreur.

`monthly` : la rotation des fichiers journaux est effectuée chaque premier jour de chaque mois.

nocompress : les fichiers ne sont pas compressés. Voir aussi compress.

nomail : les fichiers journaux ne sont transmisà aucune adresse.

nomissingok : si un fichier journal n'existe pas, logrotate emet une erreur. C'est l'option par défaut.

notifempty : si le fichier journal est vide, on n'effectue pas de rotation.

postrotate/endscript : les lignes comprises entre postrotate et endscript sont exécutées après que la rotation du fichier journal soit terminée.

prerotate/endscript : les lignes comprises entre postrotate et endscript sont exécutées avant que la rotation du fichier journal ne commence.

rotate [count] **:** les fichiers journaux subissent <count> rotation avant d'être détruits ou envoyés à l'adresse spécifiée par la directive mail. Si <count>=0 le fichier journal est détruit à chaque rotation.

size [size] **:** les fichiers journaux subissent une rotation quand leur taille dépasse la valeur de l'argument <size>. Si le suffixe de l'argument est M, l'unité sera le Mo. Si le suffixe est k l'unité sera le ko. Exemples : 100, 100k, 100M.

weekly : la rotation est effectuée chaque début de semaine.

6.2. LES EXPRESSIONS REGULIERES.

En premier lieu, nous allons étudier les 2 caractères spéciaux que sont ^ et $. Le premier sert à définir le début d'une chaîne, le second la fin :

"**^Le**": pour définir les chaînes commençant par "Le";
"**après lui$**": pour définir les chaînes se terminant par "après lui";
"**^abc$**": une chaîne qui commence et se termine par "abc" -- c'est "abc" elle-même !
"**notice**": toute chaîne contenant "notice".

Nous voyons dans ce dernier exemple que si l'on n'utilise pas les caractères ^ et $, cela indique que le motif (pattern) peut se trouver n'importe où dans la chaîne.

Nous disposons aussi des symboles *, +, et ? pour exprimer **le nombre de fois où un caractère ou une séquence de caractères peut apparaître dans la chaîne (occurrence).**

Voici leur signification :

- ***** zéro occurrence ou plus.
- **+** une occurrence ou plus.
- **?** zéro ou une occurrence.

Quelques exemples :

"**ab***": les chaînes contenant un a suivi de zéro, un, ou plusieurs b ("a", "ab", "abbb", etc.);

"**ab+**": idem, mais avec au moins 1 b derrière le a ("ab", "abbb", etc.);

"**ab?**": il doit y avoir un b ou non après le a;

"**a?b+$**": éventuellement un a (et un seul) suivi d'un ou plusieurs b, le tout en fin de chaîne.

Les **accolades { }** permettent d'indiquer plus précisément le **nombre d'occurrences** possibles :

"**ab{2}**": les chaînes contenant un a suivi d'exactement 2 b ("abb");

"**ab{2,}**": les chaînes contenant au moins 2 b consécutifs ("abb", "abbbb", etc.);

"**ab{3,5}**": les chaînes pouvant contenir de 3 à 5 b consécutifs ("abbb", "abbbb", or "abbbbb").

Notez que vous devez obligatoirement spécifier la première valeur de l'intervalle (i.e, la syntaxe "{0,2}" est correcte, mais pas "{,2}").

Vous aurez remarqué aussi que les symboles *, +, et ? ont la même signification que les intervalles "{0,}", "{1,}", et "{0,1}".

Pour définir une **séquence de caractères**, il suffit d'utiliser les **parenthèses** ( ) :

"**a(bc)***": les chaînes ayant un a suivi de zéro, une ou plusieurs fois de la chaîne "bc";

"**a(bc){1,5}**": a suivi de 1 à 5 fois de "bc".

Nous disposons aussi du **symbole |** comme opérateur booléen **OU** :

"**hi|hello**": les chaines contenant "hi" ou "hello";

"**(b|cd)ef**": les chaines contenant "bef" ou "cdef";

"**(a|b)*c**": les chaînes contenant une série de a et de b puis un c;

Le **point .** représente un **caractère unique** :

"**a.[0-9]**": pour un a suivi d'un caractère puis d'un chiffre;

"**^.{3}$**": représente les chaînes d'exactement 3 caractères.

Les **crochets [ ]** permettent eux de spécifier quels sont les **caractères autorisés** à un endroit précis de la chaîne :

"**[ab]**": les chaînes contenant un a ou un b (équivalent à "a|b");

"**[a-d]**": les chaînes contenant les caractères minuscules a à d (équivalent à "a|b|c|d" ou "[abcd]");

"**^[a-zA-Z]**": les chaînes commençant par une lettre;

"**[0-9]%**": pour un chiffre suivi du symbole % (pourcentage);

"**,[a-zA-Z0-9]$**": les chaînes se terminant par une virgule suivie d'un caractère alphanumérique.

Vous pouvez aussi indiquer les **séquences que vous voulez exclure** en utilisant le **symbole ^ en premier caractère d'une expression entre crochets [ ]** (i.e., "**%[^a-zA-Z]%**" pour les chaînes contenant un caractère qui n'est pas une lettre entre 2 symboles %).

Pour pouvoir utiliser dans une expression régulière les caractères spéciaux "^.[$()|*+?{\" en tant que caractères 'normaux', il faut les faire précéder d'un antislash \.

Comme l'antislash \ est lui-même un caractère spécial, il faut aussi le faire précéder d'un antislash \ !

Toutefois, sachez que les expressions entre crochets ne répondent pas à cette règle.

Entre crochets, tous les caractères spéciaux, y compris l'antislash, perdent leur 'pouvoir spécial' (i.e., "**[*\+?{}.]**" correspond aux caractères *****, \, +, etc.).

Pour inclure le caractère **]** dans la liste, placez le en premier (après éventuellement un ^). Pour inclure le caractère **[**, placez en première ou dernière position."